COSC4010/5010: Introductory Project Report

Fiona Moss
Computer Science Department
University of Wyoming
Laramie, Wyoming 82070
Email: fmoss1@uwyo.edu

***Abstract*—In this paper, we apply machine learning approaches to automatically detect Phishing websites. We first evaluate and compare the detection accuracy of nine classic supervised learning methods and two classic unsupervised learning methods. We then evaluate these methods under an imbalanced data setting, and show most methods suffer performance degeneration under F-measure. We finally propose remedies for degraded methods, and show they improve performance.**

I. Introduction

Phishing is the attempt to obtain sensitive information such as usernames, passwords, and credit card details (and, indirectly, money), often for malicious reasons, by disguising as a trustworthy entity in an electronic communication. A phishing website (sometimes called a "spoofed" site) tries to steal your account password or other confidential information by tricking you into believing you're on a legitimate website. You could even land on a phishing site by mistyping a URL (web address).

Damages: Phishing wrecks lives. It also steals identity.

It is estimated that between May 2004 and May 2005, approximately 1.2 million computer users in the United States suffered losses caused by phishing, totaling approximately $929 million USD. U.S. businesses lose an estimated $2 billion USD a year as their clients become victims.

In the United Kingdom losses from web banking fraud mostly from phishing almost doubled to €23.2m in 2005, from €12.2m in 2004, while 1 in 20 users claimed to have lost out to phishing in 2005.

Criminal organizations around the world use phishing to extract information from innocent citizens in order to access their bank details, steal identities, launder money and more.

Attempts can be difficult to spot with an untrained eye, and successful phishing affects everybody, from the bank manager to small children whose school, club or church group may be caught out by this type of scam.

The effect of phishing on the economy is also powerful but rarely as long lasting, hard-hitting or just downright embarrassing as when they con you.

Example: John Podestas Email Hacked: On March 19th, 2016, John Podesta (Hillary Clintons Campaign Chairman) received an email from Google. The email said someone in Ukraine had his password and tried to sign into his account. The IT team at the campaign confirmed the email was real and provided a Google specific link to change the password (and suggested he set up two-factor authentication).

Apparently, rather than using the Google link, the password change was initiated from the original phishing email, and Mr. Podestas account was compromised. That phishing attack setup a major email release by WikiLeaks something that may have contributed to Hillary Clintons loss to Donald Trump in the U.S. Presidential election.

Phishing affects companies too. The most common mistake that companies make that lead them to fall victim to phishing is to not provide adequate training to employees to maintain information security. This leads to careless internet browsing due to lack of security awareness. (https://digitalguardian.com/blog/phishing-attack-prevention-how-identify-avoid-phishing-scams)

The following are a few steps to detect phishing:

- **Website Filtering:** One approach of phishing prevention concentrates on filtering websites when they are rendered in a web browsers. In Mozilla Firefox, for instance, each web page requested by a user is checked against a blacklist of known phishing sites. This list is automatically downloaded to the local machine and updated in regular intervals.
- **Email Filtering:** Phishing emails usually contain specific phrases asking users to submit information or to access the phishing website. In conjunction these phrases often may be used as indicators for phishing and can be detected by filtering the content of emails.
- **Features for Email Data:** The features in emails can serve as input to our classifiers. We start with basic features and continue with new advanced features based on dynamic Markov chains and latent topic models. Since most of the time dubious emails cause users to enter into a phishing website, taking the features of an email into consideration to detect its genuineness can help prevent the issue by separating spam mails from genuine mails and detecting anomalies.
- **Dynamic Markov Chain Features:** Dynamic Markov Chain features are based on information theory and capture the likelihood of a message belonging to a specific class. These likelihoods as well as class membership indicators are extracted as features for the classification system. The dynamic Markov chain generation is a technique developed for arithmetic compression, the problem of compressing arbitrary binary sequences. A sequence is thought to be generated by a random source. This source can be approximated by a dynamically constructed

Markov chain. Researchers have developed a technique for the incremental construction of a Markov chain. These dynamic Markov chains have been successfully applied to text classification problems in various domains. Each class is considered as a different source, and each text belonging to the class is treated as a message emitted from the corresponding source.

- Deploying a SPAM filter that detects viruses, blank senders, etc. can prevent phishing.
- Keeping all systems current with the latest security patches and updates reduces chances of occurrence of phishing as older softwares are more vulnerable to attacks.
- Installing an antivirus solution, schedule signature updates, and monitor the antivirus status on all equipment ensures protection from malware and phishing.
- Developing a security policy that includes but isn't limited to password expiration and complexity.
- Deploying a web filter to block malicious websites.
- Encrypting all sensitive company information.
- Converting HTML email into text only email messages or disable HTML email messages.

The limitation of the above approaches is that there has to be manual effort to involve these methods. Manual efforts are tiresome and error-prone. Therefore, even lots of effort cannot ensure complete security from these attacks. Moreover, even the most up-to-date antivirus softwares sometimes fail to detect phishing websites. The above measures aid in limiting phishing to a certain extent but cannot guarantee to eliminate it and provide complete security. Thus an intelligent system may prove to be more beneficial and efficient when compared to the above methods to ensure safety from phishing.

In the paper titled "Learning to Detect Phishing Emails" published in 2006, a machine learning learning technique called PILFER was used to detect phishing emails. Subsequently in the year 2007, a paper titled "A proposal of the AdaBoost-based detection of phishing websites" was published which proved that AdaBoost, a machine learning technique, provided higher accuracy in phishing website detection when compared to other non machine learning techniques.

In the past, several machine learning techniques have been applied to detect phishing, some of them are as follows:

- **Naïve Bayes:** This classification algorithm based on applying Bayes theorem which is also known as a probabilistic algorithm. This algorithm is used in classification because of its simplicity in both during training and classifying stage. Another advantage of this algorithm is that less data is needed during training stage compared to the other machine learning based classification algorithms.
- **J48:** J48 implements Quinlan's C4.5 algorithm for creating a pruned or unpruned C4.5 decision tree. C4.5 is an augmentation of Quinlan's prior ID3 algorithm. The decision trees created by J48 can be utilized for classification. J48 constructs decision trees from an arrangement of labeled training data utilizing the idea of data entropy.
- **Support Vector Machine:** This is a well known machine learning based classification algorithm. Support Vector Machine (SVM) is based on the concept of decision planes that define decision boundaries. The decision plane also known as hyperplane separates a set of objects having different class memberships.
- **Neural Net:** A neural system is organized as an arrangement of interconnected indistinguishable units (neurons). The interconnections are utilized to send signals from one neuron to the next. Also, the interconnections have weights to upgrade the conveyance among neurons. The neurons are not capable by them-selves; in any case, when associated with others they can perform complex calculations. Weights on the interconnections are overhauled when the system is prepared, consequently significant interconnection assumes more part amid the testing stage.
- **Random Forest:** Random forest is a classifier that joins numerous tree predictors, where every tree relies on upon the estimations of an irregular vector inspected autonomously. Besides, all trees in the forests have the same appropriation.
- **IBK lazy classifier:** IBK is a k-nearest neighbor classifier that uses the same distance metric.

The methods that are going to be examined in this report are as follows:

1) Linear Regression (Least Square)
2) Ridge Regression
3) Lasso
4) Logistic Regression
5) SVM
6) Neural Network
7) Decision Tree
8) Random Forest
9) k-Nearest Neighbor
10) K-Means Clustering
11) Principal Component Analysis

Among these methods, there are 7 unsupervised (Linear Regression (Least Square, Ridge, Lasso), Logistic Regression, SVM, Neural Network, Decision Tree, Random Forest and k-Nearest Neighbor) and 3 unsupervised (K-Means Clustering, Agglomerative Clustering and PCA) methods.

We are going to examine whether over-fitting occurs in our project or not.

Over-fitting occurs when a model is excessively complex, such as having too many parameters relative to the number of observations. A model that has been over-fitted has poor predictive performance, as it overreacts to minor fluctuations in the training data. As over-fitting causes data to produce erratic results, it increases model complexity by learning errors instead of true patterns and showing the training error to be almost zero.

The dataset that we are about to use for this project is the UCI public Phishing website dataset.

II. Methodology

A. Linear Regression (Least Square)

Linear regression is a linear approach for modeling the relationship between a scalar dependent variable y and one or more explanatory variables(or independent variables) denoted X. Least Square is a method to learn Linear regression. Linear regression gives a linear relation between features and label, with the form:

$f(x) = x^T\beta + \beta_0$

where x is an example, represented by a p-dimensional feature vector, β is a p-dimensional vector of regression coefficients and β_0 is a bias term. β, β_0 are unknown parameters to be learned. Hence, "learn linear regression" = "estimate β, β_0.

Least square learns a that minimizes the following loss function:

$l(\beta) = \sum_{i=1}^{n}(x_i^T\beta - y_i)^2 = ||X.\beta - Y||_F^2$

The matrix description of Least Square method is as follows:

$l(\beta, \beta_0) = ||X\beta + \beta_0 - Y||_F^2$

Least square has a closed-form solution(analytic solution) for the linear regression model:

$\hat{\beta} = (X^TX)^{-1}X^TY$

B. Ridge Regression

Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity. When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors.

Ridge regression learns a (β, β_0) that minimizes the following loss function:

$l(\beta, \beta_0) = \sum_{i=1}^{n}(x_i^T\beta + \beta_0 - y)^2 + \lambda.\sum_{j=1}^{p}\beta^2(j)$

The matrix description of Ridge Regression is as follows:

$l(\beta, \beta_0) = ||X\beta + \beta_0 - Y||_F^2 + \lambda.||\beta||_F^2$

The closed form solution of ridge regression is as follows:

$\hat{\beta} = (X^TX + \lambda.I)^{-1}X^TY$

where columns of X are considered centered, Y is assumed centered

Hyper-parameters are parameters whose values are set prior to the commencement of the learning process. here λ is the hyper-parameter. λ shrinks the coefficient β. Features corresponding to zero coefficients are not used by the model. Hence, fewer features used decreasing the model complexity.

C. Lasso

Lasso (least absolute shrinkage and selection operator) (also Lasso or LASSO) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces.

Lasso learns a (β, β_0) that minimizes the following loss function:

$l(\beta, \beta_0) = \sum_{i=1}^{n}(x_i^T\beta + \beta_0 - y)^2 + \lambda.\sum_{j=1}^{p}|\beta(j)|$

The matrix description of Lasso is as follows:

$l(\beta, \beta_0) = ||X\beta + \beta_0 - Y||_F^2 + \lambda.||\beta||_1$

There is no closed form solution for Lasso.

The L1 regularization term is given as: $\sum_{j=1}^{p}|\beta_{(j)}|$

Hyper-parameters are parameters whose values are set prior to the commencement of the learning process. here λ is the hyper-parameter. λ shrinks the coefficient β. Features corresponding to zero coefficients are not used by the model. Hence, fewer features used decreasing the model complexity.

D. Logistic Regression

Logistic regression, or logit regression, or logit model is a regression model where the dependent variable (DV) is categorical.

Logistic regression gives the probability of x belonging to class y, as:

$p(y|x) = \frac{1}{1+e^{-y.(x^T\beta)}}$

where y is a discrete variable.

The loss function is given by:

$l(\beta) = \Pi_{i=1}^{n}p(y_i|x_i) = \Pi_{i=1}^{n}\frac{1}{1+e^{-y_i(X_i^T\beta)}}$

E. SVM

support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

If the training data are linearly separable, we can select two parallel hyperplanes that separate the two classes of data, so that the distance between them is as large as possible. The region bounded by these two hyperplanes is called the "margin".

Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors.

Here is the mathematical description of learning (soft-margin) linear SVM:

$min_{\beta,\beta_0}\frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N}\xi_i$

subject to the loss function, $\xi_i \geq 0; y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \forall i$

Here C is the hyper-parameter. Larger C gives less tolerance.

F. Neural Network

A neural network is a system of hardware and/or software patterned after the operation of neurons in the human brain. Neural networks – also called artificial neural networks – are a variety of deep learning technologies.

There are different types of activation functions(linear activation function and non-linear activation function) to introduce non-linearity such as:

- $f(x) = x$
- $f(x) = \frac{1}{1+e^{-x}}$
- $f(x) = \frac{x}{1+|x|}$
- $f(x) = \tan^{-1}(x)$

The hyper-parameters are the number of neurons per layer. The optimal number of neurons varies from problem to problem. But generally, very less neurons give less accuracy whereas a very large number of neurons cause over-fitting.

G. Decision Tree

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

Pruning: Pruning is the process of cutting of a subtree under an internal node to see if accuracy is improved.

Purpose of pruning: It is used to avoid over-fitting.

Here each node has a question on features which are considered as hyper-parameters. We need to design good questions to get a good accuracy score.

H. Random Forest

Random Forest is a bagging version of decision tree, with additional features. When learning a decision tree from one bootstrap sample, random forest:

- does not prune the tree.
- for each node question design, only uses a random subset of features.

Bootstrap is any statistical method using random subsets of available sample. Bootstrap creates subsets by sampling with replacement and each subset is called a bootstrap sample.

Here each node has a question on features which are considered as hyper-parameters. We need to design good questions to get a good accuracy score.

I. k-Nearest Neighbor

k-Nearest neighbor of a point x is a set of other k points with smallest distances to x.

The hyper-parameter is k in this case. The value of K decides the accuracy of the model.

J. K-means Clustering

K-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

The hyper-parameter is k in this case. The value of K decides the accuracy of the model.

K. Principal Component Analysis (PCA)

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components (or sometimes, principal modes of variation).

III. Experiment

A. Dataset Description

The name of the dataset that we are using is the "Phishing websites dataset" which was found on the UCI Machine learning repository.

Since we are trying to differentiate between normal websites and phishing websites, this is a classification problem. Also since we will have two classes, one for normal websites and the other for phishing websites, this is a binary class problem.

The task here is to predict whether a website is a phishing website or not based on the features of various websites provided to us.

The number of instances of this dataset according to the website it was retrieved from is 2456. Also, the total number of features in this dataset are 30.

These 30 features can be broadly classified as follows:

- Address Bar based features: IP address, favicon, etc.
- Abnormal based features: Request URL, Abnormal URL, etc.
- HTML and Javascript based features: Website forwarding, disabling right click, etc.
- Domain based features: DNS record, PageRank, etc.

This dataset contains numerical features.

Also since the dataset provided to us is a sanitized one, there is no missing data in this dataset. Also, it simplified and contains no categorical features.

B. A Coarse Performance Evaluation

In this section, we evaluate performance of seven supervised learning methods, two clustering methods and two feature extraction methods using random (or default, or optimal) hyper-parameter settings. We randomly select 75% examples in the data set for training, and use the rest 25% for testing.

First, we apply seven supervised learning methods to learn prediction models from the training sample, and evaluate prediction accuracies of these models on the testing sample. Results are presented in Table III-B.

Then, we apply PCA to reduce feature dimensionality, plot sample distributions in the new feature space and evaluate prediction performance in the new feature space.

We also examine the impact of PCA on prediction accuracy. To do so, we first apply PCA to reduce data dimensionality to 15, and then learn a SVM model. The prediction accuracy of this model is in Table III-B

We apply KMeans and Agglomerative clustering on the training sample, with K = 2. We plot the clustering result in the new PCA feature space.

TABLE I
TESTING PERFORMANCE OF ALL ALGORITHMS

Method	Prediction Accuracy
Linear Regression (Least Square)	92.510853835%
Ridge Regression	92.510853835%
Lasso	55.8972503618%
Logistic Regression	92.8726483357%
SVM	94.3921852388%
Neural Network	95.7308248915%
Decision Tree	95.4052098408%
Random Forest	96.8162083936%
k-Nearest Neighbor	95.2604920405%

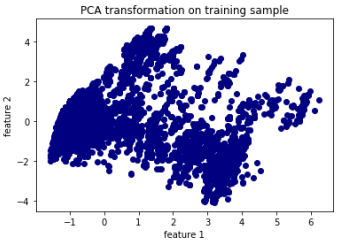

Fig. 1. Training Sample Distribution based on PCA Directions.

TABLE II
TESTING PERFORMANCE OF ALL ALGORITHMS

Method	Prediction Accuracy
PCA (15) + SVM	94.0303907381%
PCA (15) + Linear	91.8234442836%
PCA (15) + Ridge	91.8234442836%
PCA (15) + Lasso	54.8118668596%
PCA (15) + Logistic	92.2937771346%
PCA (15) + Neural Networks	96.3458755427%
PCA (15) + Decision Trees	96.1649782923%
PCA (15) + Random Forest	96.8523878437%
PCA (15) + KNN	95.8031837916%
KPCA (15) + SVM	92.8726483357%
LDA (1) + SVM	94.0303907381%

TABLE III
CLUSTERING PERFORMANCE OF TWO ALGORITHMS

Method	Measure
KMeans	0.00254871501204 (Adjusted Rand Index)
Agglomerative	-0.00255104435661 (Adjusted Rand Index)
KMeans	1265.37923417 (Calinski-Harabaz Index)
Agglomerative	1153.49652792 (Calinski-Harabaz Index)

C. Sensitivity Analysis, Over-Fitting and Model Selection

We first evaluate the testing accuracy of Lasso versus different choices of L-1 regularization coefficient λ in (II-C). The experimental setting remains the same as in previous section. Results are shown in Figure III-C.

We also examine the regression coefficients of Lasso when $\lambda \in \{1, 0.1, 0.01\}$.

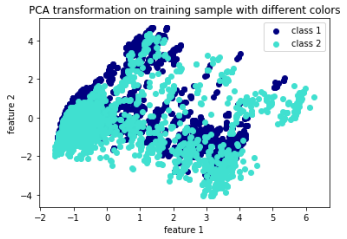

Fig. 2. Colored Training Sample Distribution based on PCA Directions.

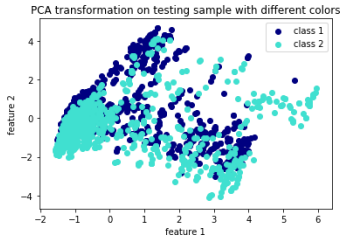

Fig. 3. Colored Testing Sample Distribution based on PCA Directions.

TABLE IV
CLUSTERING PERFORMANCE OF TWO ALGORITHMS

Hyper-Parameter λ	Testing Accuracy
1	0.556942794567 (2.12881035105e-05)
0.1	0.882405820819 (6.82081871246e-06)
0.01	0.913341640334 (2.36028319954e-05)

Since, the accuracy score of 0.01 is the best among all the values of the hyper-parameter, it is the best choice among 1, 0.1 and 0.01.

D. Acknowledgements

I was absent due to a conference and was lagging behind in class. Asma helped me by showing me the code that you gave during the lab session. I looked at that code, understood it and implemented it on my own. Also, Dayanand and I came up with the idea of the solution of task 3 of experiment_eval2 together. We implemented this idea in our own separate ways. I also used some of your code that you mentioned in the comments of the project report.

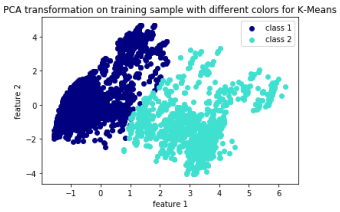

Fig. 4. Result of KMeans Clustering.

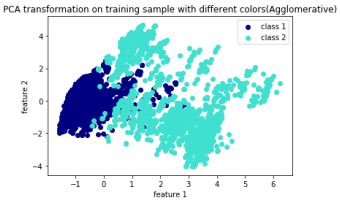

Fig. 5. Result of Agglomerative Clustering.

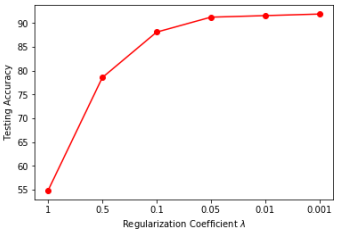

Fig. 6. Performance of Lasso versus Different λ's.

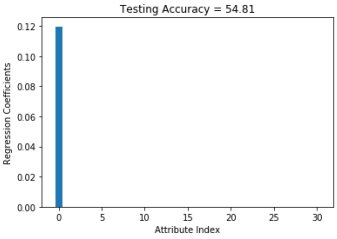

Fig. 7. Performance of Lasso with $\lambda = 1$

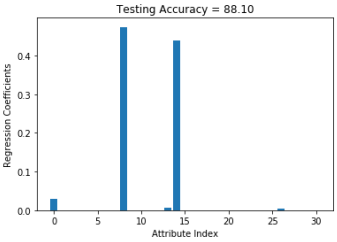

Fig. 8. Performance of Lasso with $\lambda = 0.1$

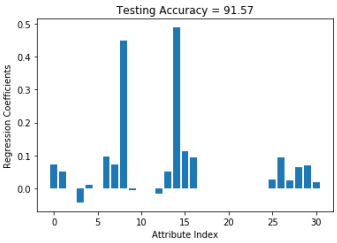

Fig. 9. Performance of Lasso with $\lambda = 0.01$

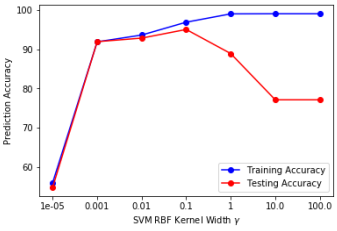

Fig. 10. Performance of SVM with RBF Kernel